UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/608,040	06/30/2003	Hajime Ogawa	2003_0866A	3923

513 7590 10/16/2007

WENDEROTH, LIND & PONACK, L.L.P.
2033 K STREET N. W.
SUITE 800
WASHINGTON, DC 20006-1021

EXAMINER	
WEI, ZHENG	

ART UNIT	PAPER NUMBER
2192	

MAIL DATE	DELIVERY MODE
10/16/2007	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Office Action Summary	Application No.	Applicant(s)	
	10/608,040	OGAWA ET AL.	
	Examiner	Art Unit	
	Zheng Wei	2192	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *31 July 2007*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-5,8-13,16-28,30,32,33,35-40 and 42-54* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☐ Claim(s) *1-5, 8-13, 16-28, 30, 32, 33,35-40, 42-54* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

Remarks

1. This office action is in response to the amendment filed on 07/31/2007.

2. Claims 6, 7, 14, 15, 29, 31, 34 and 41 have been canceled.

3. Claims 1-5, 8-12, 16-22, 24-28, 30, 32, 33, 36-40, 42 and 43 have been amended.

4. Claims 44-54 have been added.

5. The objection to specification is withdrawn in view of applicant submitted replacement abstract.

6. The objection to claims 3 and 4 is withdrawn in view of applicant's amendment

7. The 35 U.S.C. 112 second paragraph rejection of claims 2, 30 and 34 are withdrawn in view of the Applicant's amendment and cancellation of claims 34.

8. The 35 U.S.C. § 101 rejection to claims 1-4, 6-12, 14-22, 24-27, 29-34 and 36-43 is withdrawn in view of applicant's amendment and cancellation of related claims.

9. Claims 1-5, 8-13, 16-28, 30, 32, 33, 35-40 and 42-54 remain pending and have been examined.

Response to Arguments

10. Applicant's arguments filed on 07/31/2007, in particular on pages 25-41, have been fully considered but they are not persuasive. For example:

- Page 25, section IV, Claim Rejections under 35 U.S.C. § 101, the Applicant argues that claims 5, 13, 23, 28 and 35 are directed to statutory subject matter. However, the Examiner respectfully disagrees. The claims direct to a source program recorded on a computer-readable recording medium. However, the source code itself without further compilation is only a collection of characters and data. These kinds of source code programs **cannot be executed by computer** and cause computer to perform certain functions. Although the source programs include at least one of descriptions for directing a compiler to translate the source program, it is the compiler application program executed by computer to perform function to read source code including the description and direct compiler itself to translate source code. Therefore, the 35 U.S.C. § 101 rejection to claims 5, 13, 23, 28 and 35 is maintained.

Claim Rejections - 35 USC § 101

11. 35 U.S.C. 101 reads as follows:

> Whoever invents or discovers any new and useful process, machine, manufacture, or composition of matter, or any new and useful improvement thereof, may obtain a patent therefor, subject to the conditions and requirements of this title.

12. Claims 5, 13, 23, 28 and 35 are rejected under 35 U.S.C. 101 because the claimed invention is directed to non-statutory subject matter.

Claims 5, 13, 23, 28 and 35: These claims claim the source programs, which are recorded on computer-readable recording medium, wherein the source program

includes at least one of descriptions. However, source programs including descriptions themselves are "Nonfunctional descriptive material" and when nonfunctional descriptive materials are recorded on some computer-readable medium, in a computer or on electromagnetic carrier signals, they are not statutory since no requisite functionalities are present to satisfy the practical, useful application requirements. See M.P.E.P. 2106.01 (II)

Claim Rejections - 35 USC § 103

13. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

14. Claims 1-5 and 36 are rejected under 35 U.S.C. 103(a) as being unpatentable over Stallman (Richard M. Stallman, Using and Porting the GNU Compiler Collection for GCC 3.1) in view of Nakamura (Nakamura et al., Architecture and Compiler Co-Optimization for High Performance Computing) and further in view of Popovic (Popovic et al., A C Compiler Design Concept Used for MAS Family of Digital Signal Processors)

Claim 1:

Stallman discloses a compiler that translates a source program into a machine language program comprising:

- a directive acquisition unit operable to acquire a directive for optimizing a machine language program to be generated (see for example, p.9, "Optimization options") and
- an optimization unit operable to perform optimization by generating a sequence of machine language instructions following an acquired directive (see for example, see p.49, section 3.10, "Options that Control Optimization"),

But does not explicitly disclose, wherein the optimization unit performs the optimization by deciding array data allocated to a global memory region following a directive when the directive acquisition unit acquires the directive on the array data to be allocated to the global memory region. However, Nakamura in the same analogous art of compiler optimization discloses using directive to allocate array in software controllable memory (see for example, p.52, section 3, Directive-Based Compiler, also see p.53, section 3.2 Example of Directives). But neither of them explicitly discloses, wherein the optimization unit performs the optimization by deciding array data allocated to a global memory region following a directive when the directive acquisition unit acquires the directive on the array data to be allocated to the global memory region. However, Popovic in the same analogous art of compiler optimization discloses

- using directive to allocate memory resource (see for example, p.607, section II, Compiler Structure, "To enable programmers to manually allocate memory resources (to place variables at specific addresses) a new pragma directive has been introduced. This directive is called mem_alloc")
- wherein the global memory region is specified by ahead address (org:address) and a displacement (size) (see for example, p.607, last paragraph, example code, "#pragma mem_alloc(org:address,.. size:4)")
- wherein the head address is indicated by a value stored in a register (see for example, p.607, last paragraph , example code, the value of "address")
- wherein the displacement is within a range of the global memory region that can be access by one instruction

Therefore, it would have been obvious to one having ordinary skill in the art at the time the invention was made to add this specific optimization option to Stallman's gcc compiler by using Popovic's method to allocate global memory resource for array as disclosed by Nakamura. One would have been motivated to do so to improve the performance for high performance computing as suggested by Nakamura (see for example, p.51, section 2.2 Benefit of SCM). It also would have been obvious to one having ordinary skill in the art at the time the invention was make to understand that such compiler application can be run in a computing device to compile software source program to perform the function as addressed above.

Claim 2:

Stallman, Popovic and Nakamura disclose the compiler apparatus/device according to claim 1, Nakamura further discloses:

- wherein the directive acquisition unit acquires designation of a maximum data size of array data to be allocated to a global memory region together with a directive for translating the source program (see for example, p.52, Figure 2. directive "!$scm begin (<array_name>...)" and related text, also see, p.53, section Example of Directives"),

But none of them discloses

- the optimization unit, out of array data declared by the source program, allocates array data whose maximum data size does not exceed the maximum data size to a global memory region and array data whose maximum data size exceeds the maximum data size to a memory region out of the global memory region.

However, it is well known in the computer art that memory resource includes global/local memory. If the "pragma" requested array size excesses global memory size, it will allocate local memory next to the global memory address space. Therefore, it would have been obvious to one having ordinary skill in the art at the time the invention was made to use directive to specify the specific memory location where the compiler allocates the array variable to according to the predefined parameter (maximum data size). One would have been motivated

to do so to improve the performance for high performance computing as suggested by Nakamura (see for example, p.51, section 2.2 Benefit of SCM)

Claims 3-4:

Stallman, Popovic and Nakamura disclose the compiler apparatus/device according to claim 1, Nakamura further disclose:

- wherein the directive acquisition unit detects a directive for not allocating/ allocating specific array data to the global memory region in the source program (see for example, p.52, Figure 2. directive "!$scm begin (<array_name>...)" and related text, also see, p.53, section Example of Directives"), and
- the optimization unit allocates array data that are an object of a directive detected by the directive acquisition unit to a memory region out of the global memory region/global memory region (see for example, p.53, section Example of Directives")

Therefore, it would have been obvious to one having ordinary skill in the art at the time the invention was made to add this specific optimization option to Stallman's gcc compiler. One would have been motivated to do so to improve the performance for high performance computing as suggested by Nakamura (see for example, p.51, section 2.2 Benefit of SCM)

Claim 5:

Claim 5 is a software program product version of claimed apparatus discussed in claims 1-4 above, wherein all claimed limitations have been address and/or set forth above. Therefore, as the references teach all the limitation of claims 1-4, they also teach the limitations of claim 5. Thus, it also would have been obvious.

Claim 36:

Claim 36 is a software program product version of claimed apparatus discussed in claims 1-4 above, wherein all claimed limitations have been address and/or set forth above. Therefore, as the references teach all the limitation of claims 1-4, they also teach the limitations of claim 36. Thus, it also would have been obvious.

15. Claims 8 and 52 are rejected under 35 U.S.C. 103(a) as being unpatentable over Stallman (Richard M. Stallman, Using and Porting the GNU Compiler Collection for GCC 3.1) in view of Sun (Sun workshop compiler c 4.2 document: C user's guide)

Claims 8 and 52:

Stallman discloses a compiler apparatus that translates a source program into a machine language program comprising:

- a directive acquisition unit operable to acquire a directive for optimizing a machine language program to be generated (see for example, p.9, "Optimization options"); and
- an optimization unit operable to perform optimization by generating a sequence of machine language instructions following an acquired directive (see for example, see p.49, section 3.10, "Options that Control Optimization")

But Stallman does not explicitly disclose using directive to perform/not perform software pipelining optimization. However, Sun in the same analogous art of software compiler, discloses using directive for optimization on software pipelining (see for example, p.12, "#pragma pipeline(n)").

Therefore, it would have been obvious to one having ordinary skill in the art at the time the invention was made to understand that if the directive is specified other options except "pipeline(n)", certainly the directive detect unit will detect a directive other than pipelining and optimization unit will not perform pipelining optimization. It also would have been obvious to one having ordinary skill in the art at the time the invention was make to understand that such compiler application can be run in a computing device to compile software source program to perform the function as addressed above.

16. Claims 9-13, 37, 53 and 54 are rejected under 35 U.S.C. 103(a) as being unpatentable over Stallman (Richard M. Stallman, Using and Porting the GNU Compiler Collection for GCC 3.1) in view of Sun (Sun workshop compiler c 4.2 document: C user's guide) in further view of Granston (US 6,892,380)

Claims 9-10 and 53-54:

Stallman and Sun disclose the compiler apparatus according to claim 8, Stallman further discloses insert prolog and epilog portion to increase execution but neither of them discloses:

- wherein the directive acquisition unit detects a directive for performing the optimization by software pipelining that removes/does not remove a prolog portion and an epilog portion of a specific loop processing in the source program, and
- the optimization unit performs the optimization by software pipelining of loop processing that is an object of the directive detected by the directive acquisition unit whenever possible to remove the prolog portion and the epilog portion.

However, Granston in the same analogous art of software pipelining discloses a method to remove (omit) the epilog to reduce code size (see for example, p.4, lines 19-26, also see example at p.2 and related text). It is also well known in the computer art that the compilation directive can be added to selectively optimize the specific loop. One would have been motivated to compile software program more flexible.

Claim 11:

Stallman and Sun disclose the compiler apparatus according to claim 8, but neither of them discloses:

- wherein the directive acquisition unit detects designation of the number of iterations of specific loop processing in the source program, and
- the optimization unit performs optimization of loop processing that is an object of the designation detected by the directive acquisition unit based on the designated number of iterations.

However, Granston in the same analogous art of software pipelining discloses a designation of the number of iterations of specific loop (see for example, col.1, lines 29-39, "n represents the number of desired iterations"). Therefore, it would have been obvious to one having ordinary skill in the art at the time the invention was made to specifically define the number of iteration. One would have been motivated to do so to use this number compare with minimum required trip count to decide executing pipelined version or original version as indicated by Granston (see for example, col.3, lines 43-49, "if (n>= min required trip count")

Claim 12:

Stallman, Sun and Granston disclose the compiler apparatus according to claim 11, Granston further discloses:

- wherein the designation of the number of the iterations is the minimum number by which the loop processing is iterated (see for example, col.3, lines 43-49, "if (n>= min required trip count"), and
- the optimization unit performs the optimization by software pipelining when the minimum number is equivalent to or larger than the number of iterations that overlap by software pipelining (see for example, col.3, lines 43-49, "if (n>= min required trip count; pipelined version; else original version; endif" and relate description).

Therefore, it would have been obvious to one having ordinary skill in the art at the time the invention was made to specifically define the number of iteration as the minimum iterated number. One would have been motivated to do so to use this number to generate multiversion code indicated by Granston (see for example, col.3, lines 50-51, "This technique is referred to as multiversion code generation)

Claim 13:

Claim 13 claims a computer source code that is written according to requirement of compiler apparatus discussed in claims 8-10 above. Because all claimed limitations of said compiler apparatus have been address and/or set forth above in claims 8-10. Therefore, it would have been obvious to perform the feature that the directive specified while using such compiler apparatus to compile said computer source code.

Claim 37:

Claim 37 is a software program product version of claimed apparatus in claims 8-12 above, wherein all claimed limitations have been address and/or set forth above. Therefore, as the references teach all the limitation of claims 8-12, they also teach the limitations of claim 37. Thus, it also would have been obvious.

17. Claims 16, 17, 38, 44 and 45 are rejected under 35 U.S.C. 103(a) as being unpatentable over Stallman (Richard M. Stallman, Using and Porting the GNU Compiler Collection for GCC 3.1) in view of PGI (PGI Workstation User's Guide-9 Optimization Directive and Pragmas)

Claims 16, 17 and 44, 45:

Stallman discloses a compiler apparatus that translates a source program into a machine language program comprising:

- a directive acquisition unit operable to acquire a directive for optimizing a machine language program to be generated (see for example, p.9, "Optimization options"); and
- an optimization unit operable to perform optimization by generating a sequence of machine language instructions following an acquired directive (see for example, see p.49, section 3.10, "Options that Control Optimization"),

- loop unrolling option(see for example, see p.49, section 3.10, "Options that Control Optimization", and also see p.55, lines 38-42, "-funroll-loops" and related description).

but does not explicitly discloses wherein the optimization unit performs optimization by loop unrolling following a directive when the directive acquisition unit acquires the directive on the optimization by loop unrolling.

However, PGI in the same analogous art of compiler software optimization using directive discloses adding pragmas to C and C++ specific to perform or not perform loop unrolling (see for example, section 9.4 Adding Pragmas to C and C++, Table 9-2 C/C++ Pragma Summary, "unroll" and "nounroll" and related description). Therefore, it would have been obvious to one having ordinary skill in the art at the time the invention was made to also add directive to gcc compiler to selectively perform or not perform loop unrolling optimization. One would have been motivated to do so selectively perform or not perform loop unrolling optimization to specific loop instead of all software program.

Claim 38:

Claim 38 is a software program product version of claimed apparatus in claims 16, 17, 44 and 45 above, wherein all claimed limitations have been address and/or set forth above. Therefore, as the references teach all the limitation of claims 16, 17, 44 and 45, they also teach the limitations of claim 38. Thus, it also would have been obvious.

18. Claims 18-23 and 46-51 are rejected under 35 U.S.C. 103(a) as being unpatentable over Stallman (Richard M. Stallman, Using and Porting the GNU Compiler Collection for GCC 3.1) in view of PGI (PGI Workstation User's Guide-9 Optimization Directive and Pragmas) and further in view of Geva (Robert Y. Geva, US 6,539,541)

Claim 18:

Stallman discloses a compiler apparatus that translates a source program into a machine language program comprising:

- a directive acquisition unit operable to acquire a directive for optimizing a machine language program to be generated (see for example, p.9, "Optimization options"); and
- an optimization unit operable to perform optimization by generating a sequence of machine language instructions following an acquired directive (see for example, see p.49, section 3.10, "Options that Control Optimization"),
- loop unrolling option(see for example, see p.49, section 3.10, "Options that Control Optimization", and also see p.55, lines 38-42, "-funroll-loops" and related description).

but does not explicitly discloses wherein the optimization unit performs optimization by loop unrolling following a directive when the directive acquisition unit acquires the directive on the optimization by loop unrolling.

However, PGI in the same analogous art of compiler software optimization using directive discloses adding pragmas to C and C++ specific to perform or not perform loop unrolling (see for example, section 9.4 Adding Pragmas to C and C++, Table 9-2 C/C++ Pragma Summary, "unroll" and "nounroll" and related description). Therefore, it would have been obvious to one having ordinary skill in the art at the time the invention was made to also add directive to gcc compiler to selectively perform or not perform loop unrolling optimization. One would have been motivated to do so selectively perform or not perform loop unrolling optimization to specific loop instead of all software program.

But neither of them discloses detecting a directive of designation of the number of iteration. However, Geva in the same analogous art of loop unrolling discloses the number of iterations defined by the directive (see for example, col.9, lines 37-38, "where the value of loop iterations is read by the program from an input file"). Therefore, it would have been obvious to one having ordinary skill in the art at the time the invention was made to specify the number of iterations of specific loop processing in the source program. One would have been motivated to do so to ensure the number of iterations can be determined at compile time and further guarantee the loop unrolling can be performed as suggest by Geva (see for example, col.9, lines 31-42).

Claim 19:

Stallman, PGI and Geva disclose the compiler apparatus according to claim 18, Geva further discloses: the optimization unit restrains generation of an escape code that is needed in the case of the number of the iterations being 0 when the minimum number is 1 or more (see for example, col.10, lines 9-10, "When the unrolled loop is a counted loop, there is no need to test for the exit condition inside the unrolled body.").

Claim 20:

Stallman, PGI and Geva disclose the compiler apparatus according to claim 18, Geva further disclose the optimization unit performs the optimization by loop unrolling when the minimum number is equivalent to or more than the number of development by the loop unrolling (see for example, col.10, lines 6-9, "One such optimization may be loop unrolling where a loop is unrolled 'n' times, such that 'n-1' additional copies of the loop body are made").

Claims 21 and 22:

Stallman, PGI and Geva disclose the compiler apparatus according to claim 18, neither of them explicitly disclose the number of iteration is even/odd number. However, Geva in the same analogous art of loop unrolling discloses the number of iterations defined by the directive (see for example, col.9, lines 37-38, "where the value of loop iterations is read by the program from an input file"). Therefore, it would have been obvious to one having ordinary skill in the art at the time the

invention was made to specify the number of iterations of specific loop processing in the source program. One would have been motivated to do so to ensure the number (even/odd number) of iterations can be determined at compile time and further guarantee the loop unrolling can be performed as suggest by Geva (see for example, col.9, lines 31-42).

Claim 23:

Claim 23 claims a computer source code that is written according to requirement of compiler apparatus discussed in claims 16 and18 above. Because all claimed limitations of said compiler apparatus have been address and/or set forth above in claims 16 and 18. Therefore, it would have been obvious to perform the feature that the directive specified while using such compiler apparatus to compile said computer source code.

Claims 46-51 are other version of device claims performing the similar claimed method as in claims 18-22 addressed above, wherein all claimed limitation functions have been addressed and/or set forth above and certainly a computer system would need to run and/or practice such function steps disclosed by reference above. Thus, they also would have been obvious.

19. Claims 24-28 are rejected under 35 U.S.C. 103(a) as being unpatentable over Stallman (Richard M. Stallman, Using and Porting the GNU Compiler Collection

for GCC 3.1) in view of Faraboschi (Faraboschi, The Latest Word in Digital and media Processing)

Claim 24:

Stallman discloses a compiler apparatus that translates a source program into a machine language program comprising:

- a directive acquisition unit operable to acquire a directive for optimizing a machine language program to be generated (see for example, p.9, "Optimization options"); and
- an optimization unit operable to perform optimization by generating a sequence of machine language instructions following the acquired directive (see for example, see p.49, section 3.10, "Options that Control Optimization"),

but does not explicitly disclose, wherein the optimization unit performs optimization on an "if" conversion following a directive when the directive acquisition unit acquires the directive on the "if" conversion. However, Faraboschi in the same analogous art of compiler for software optimization discloses converting Ifs to SELECT (see for example, p.76, section Converting Ifs to SELECTS, "As an alternative to predication, a compiler can usually convert simple IF into something called SELECT operations"). But Faraboschi still does not disclose using directive to do such optimization. However, it is well known such compiler optimization is also can be implement the form of directive/selective option instead of applying for all software programs. One

would have been motivated to do so to selective perform optimization in the case of branch operation becoming a pipelined operation which can cause significantly more complexity in the compiler as addressed by Faraboschi (see for example, p.73-74, section Branch Architecture)

Claims 25-27:

Stallman and Faraboschi disclose the compiler apparatus according to claim 24, but no one explicitly discloses conditional compilation and optimization by making/not making conversion. However, Prata in the same analogous art of Prata also discloses:

- wherein the directive acquisition unit detects a directive for making/not making an "if" conversion of a specific "if" structure sentence in the source program (see for example, p.577-581, Conditional Compilation), and
- the optimization unit restrains/makes the "if" conversion of all "if" structure sentences in the source program when the directive acquisition unit acquires the directive (#define, #ifdef, #endif) for not making/making the "if" conversion (see for example, p.577-581, Conditional Compilation)

Therefore, it would have been obvious to one having ordinary skill in the art at the time the invention was made to combine Stallman's compilation directive option with Prata's conditional compilation directive or directly use Faraboschi and Stallman's gcc compiler to compile source code containing Prata's

conditional compilation directive to make/not make "if" conversion as taught by Prata (see for example, p.577, section "Conditional Compilation", "You can use them to tell the compiler to accept or ignore blocks of information or code according to conditions at the time of compilation.")

Claim 28:

Claim 28 claims a computer source code that is written according to requirement of compiler apparatus discussed in claims 24-27 above. Because all claimed limitations of said compiler apparatus have been address and/or set forth above in claims 24-27. Therefore, it would have been obvious to perform the feature that the directive specified while using such compiler apparatus to compile said computer source code.

Claim 39:

Claim 39 is a software program product version of claimed apparatus in claims 24-27 above, wherein all claimed limitations have been address and/or set forth above. Therefore, as the references teach all the limitation of claims 24-27, they also teach the limitations of claim 39. Thus, it also would have been obvious.

Conclusion

20. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure.

21. Applicant's arguments with respect to claims rejection have been considered but are moot in view of the new grounds of rejection.

Applicant's amendment necessitated the new ground(s) of rejection presented in this Office action. Accordingly, **THIS ACTION IS MADE FINAL**. See MPEP § 706.07(a). Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the mailing date of this final action.

22. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Zheng Wei whose telephone number is (571) 270-1059 and Fax number is (571) 270-2059. The examiner can normally be reached on Monday-Thursday 8:00-15:00.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Tuan Q. Dam can be reached on (571) 272-3695. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Any inquiry of a general nature of relating to the status of this application or proceeding should be directed to the TC 2100 Group receptionist whose telephone number is 571- 272-1000.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

ZW

TUAN DAM
SUPERVISORY PATENT EXAMINER